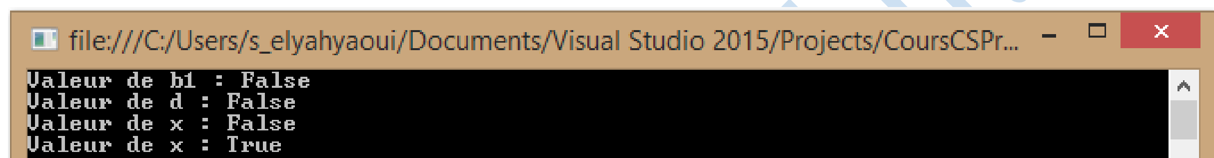
Exemple

```
double x1, y1;
x1 = 1.5; y1 = 1.25;
bool b1 = x1 <= y1;
Console.WriteLine("Valeur de b1 : " + b1);

bool a = true, b = !a, c = false, d = c & b;
Console.WriteLine("Valeur de d : " + d);
bool x = b ^ c;
Console.WriteLine("Valeur de x : " + x);
x = (b ^ c) || a;
Console.WriteLine("Valeur de x : " + x);

Console.ReadKey();
```



REMARQUE

L'opérateur **&** vérifie **les deux booléens**, même quand le **1ier** est **faux**. Tandis que l'opérateur **&&** vérifie le **1ier** booléen, et s'il est **faux**, il retourne la valeur ***false*** **sans vérifier le 2ième**.

Cela est valable aussi pour **||** et **|**

Exercice

Donner des valeurs aux variables **i**, **j** et **k** pour que les variables **a**, **b** et **c** reçoivent toutes la valeur **true** :

```
{
  bool i = …, j = …, k = …;

  bool a = (i && k) ^ (j || i);


  i = …; j = …; k = …;

  bool b = (i ^ !j) && (!k && j);


  i = …; j = …; k = …;

  bool c = (i || true) && ((j & k) && i);
}
```

i	j	k
…	…	…
…	…	…
…	…	…

Exercice

Donner la valeur affichée de la variable **d**.

```
{
  int a, b, c, d;

  a = 0;
  b = a++;
  c = a++;
  d = a * ++c * ++b;
  Console.WriteLine("valeur de d : " + d);

  a = 10;
  b = --a + c;
  c += b;
  d = c += a++;
  Console.WriteLine("valeur de d : " + d);

  d = c += ++a;
  Console.WriteLine("valeur de d : " + d);
  Console.ReadKey();
}
```

5. La conversion entre les types

1. Conversion entre les chiffres

Définition : Transtypage

Le ***transtypage*** (on dit aussi ***conversion***, ***cast***, ***casting***), est l'action de transformer une valeur d'un type A à un autre type B.
Ces types doivent être compatibles, c.à.d. de la même famille. (entiers, réels, …).

Exemples

```
short a = 200;
int b = a;

b = 1000;
// a = b; // erreur de compilation

a = (short)b;
Console.WriteLine("conversion explicite 1 : " + a);

b = 50000;
a = (short)b;
Console.WriteLine("conversion explicite 2 : " + a);

int x = 1000;
double y = x;
Console.WriteLine("conversion implicite 2 : " + y);

// x = y; // erreur de compilation

x = (int)y;
Console.WriteLine("conversion explicite 3 : " + x);

y = 15000.5526;
x = (int)y;
Console.WriteLine("conversion explicite 4 : " + x);

Console.ReadKey();
```

Conversion **implicite**, réalisée automatiquement par le compilateur, du type **short** vers le type **int**

Conversion du type **int** (plus grand) vers le type **short** (plus petit). Donc erreur de compilation. (Même si la valeur peut être stockée dans **a**)

Conversion **explicite**

Conversion explicite, avec **débordement**

Même erreur de la conversion précédente

Conversion explicite, avec **perte de la virgule flottante**

file:///C:/Users/s_elyahyaoui/Documents/Visual Studio 2015/Projects/CoursCSPr...

```
conversion explicite 1 : 1000
conversion explicite 2 : -15536
conversion implicite 2 : 1000
conversion explicite 3 : 1000
conversion explicite 4 : 15000
```

2. Conversion entre les chaines de caractères et les chiffres

Définition : Conversion chaine → chiffre

Convertir une chaine de caractères en un chiffre, signifie extraire les chiffre que contient cette chaine.

On peut faire cette action de deux manières possibles :

- En utilisant la fonction ***Parse()*** des types int, short, byte, long, double, float, decimal et char.

Syntaxe :

```
type x = type.Parse(une_chaine_de_caracteres)
```

Exemple :

```
int x = int.Parse("10");
```

- En utilisant les fonctions de la classe **Convert** (***ToBoolean, ToByte, ToChar, ToDecimal, ToDouble, ToInt16, ToInt32, ToInt64, ToUInt16,*** …).

Exemple :

```
int x = Convert.ToInt16("10");
```

Exemples

```
string s1 = "15"; int i;
// i = s1; // erreur de compilation

i = Convert.ToInt16(s1);
Console.WriteLine("valeur de i : " + i);

s1 = "122";
i = int.Parse(s1);
Console.WriteLine("valeur de i : " + i);

s1 = "1450,663";
double d = Convert.ToDouble(s1);
Console.WriteLine("valeur de d : " + d);

Console.ReadKey();
```

Types incompatibles

Fonctions `ToInt16`, `ToInt32`, `ToInt64` de la classe **`Convert`**

Fonction `Parse` du type **`int`**, même rôle que les fonctions `ToInt` de la classe **`Convert`**

Fonctions `ToDouble` de la classe **`Convert`**. Ou bien la fonction **`Parse`** du type **`double`** : **`double d = double.Parse(s1);`**

file:///C:/Users/s_elyahyaoui/Documents/Visual Studio 2015/Projects/CoursCSPr...

```
valeur de i : 15
valeur de i : 122
valeur de d : 1450,663
```

REMARQUES

- Lors de la conversion d'une chaine de caractères vers un chiffre (réel ou entier), la chaine ne doit contenir aucun caractère alphabétique ou autre.
- Lors de la conversion d'une chaine de caractères vers un réel, la chaine doit avoir le format suivant : **"xxx,xxx"** au lieu de **"xxx.xxx"**.
- Pour le compilateur C#, les réels sont par défaut des « **doubles** », donc pour utiliser des « float », il faut faire une **conversion explicite** de la valeur réelle vers le type float.
  1ière Syntaxe :
  `float x = xxx.xxxF` ou `xxx.xxxf`
  2ième Syntaxe :
  `float x = (float)xxx.xxx`

Exemples

```
string s1 = "1450.663";
double d = Convert.ToDouble(s1);
Console.WriteLine("valeur de d : " + d);

s1 = "18ans";
int age = int.Parse(s1);
Console.ReadKey();

float f1 = 11.25;

float f2 = 11.25F;
```

- `double d = Convert.ToDouble(s1);` : Erreur lors de l'exécution, format de la chaine incorrecte
- `int age = int.Parse(s1);` : Erreur lors de l'exécution, format de la chaine incorrecte
- `float f1 = 11.25;` : Erreur de compilation, type **`double (11.25)`** plus grand que le type **`float`**
- `float f2 = 11.25F;` : Conversion explicite en rajoutant le caractère **`f`** ou **`F`** à la fin. On bien : **`float f2 = (float)11.25;`**

6. Lire des informations depuis le clavier

1. Fonction ReadLine

Définition

La fonction ***ReadLine( )*** sert à lire **toute une ligne** tapée au clavier, **sous forme d'une chaine de caractères**.
Cette fonction bloque l'exécution du programme jusqu'à ce que l'utilisateur tape la touche **Entrée**. (Si l'utilisateur n'écrit rien, alors la chaine lue est la chaine vide)

Exemple 1 : Lire une chaine de caractères

```
Console.Write("Bonjour, tapez votre adresse : ");
string adr = Console.ReadLine();
Console.Write("L'adresse que vous avez tapee est : " + adr);
Console.ReadKey();
```

```
file:///C:/Users/s_elyahyaoui/Documents/Visual Studio 2015/Projects/CoursCSPr...
Bonjour, tapez votre adresse : Rabat, Maroc
L'adresse que vous avez tapee est : Rabat, Maroc_
```

Exemple 2 : Lire un chiffre entier

```
Console.WriteLine("Bonjour, tapez la quantite stock du produit : ");
string ligne = Console.ReadLine();
int qte = int.Parse(ligne);
Console.Write("La quantite que vous avez tapee est : " + qte);
Console.ReadKey();
```

```
file:///C:/Users/s_elyahyaoui/Documents/Visual Studio 2015/Projects/CoursCSPr...
Bonjour, tapez la quantite stock du produit :
100
La quantite que vous avez tapee est : 100
```

Exemple 3 : Lire un chiffre réel

```
Console.WriteLine("tapez la quantite commandee du produit : ");
string ligne = Console.ReadLine();
int qte_comm = int.Parse(ligne);
Console.WriteLine("tapez le prix unitaire du produit : ");
ligne = Console.ReadLine();
double prix_unit = Convert.ToDouble(ligne);
Console.Write("\nLe prix total est : " + qte_comm * prix_unit);
Console.ReadKey();
```

```
file:///C:/Users/s_elyahyaoui/Documents/Visual Studio 2015/Projects/CoursCSPr...
tapez la quantite commandee du produit :
5
tapez le prix unitaire du produit :
12,25

Le prix total est : 61,25
```



IMPORTANT

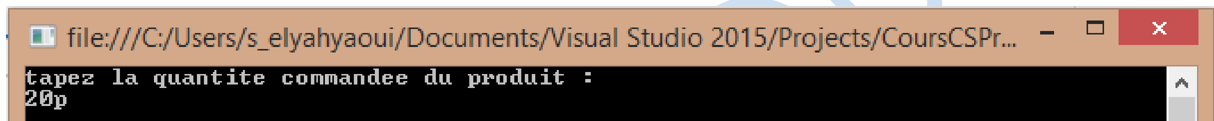
Lors de la lecture d'un chiffre depuis le clavier, la valeur tapée par l'utilisateur peut être incorrecte (ex. 11.25 au lieu de 11,25), ou contenir un caractère (ex. 11, 25). Cela produira une erreur lors de l'exécution au moment de la tentative de conversion. Ce type d'erreur est appelé « exception » (*les exceptions seront traitées dans le chapitre 3*).

Exemple

```
Console.WriteLine("tapez la quantite commandee du produit : ");
string ligne = Console.ReadLine();
int qte_comm = int.Parse(ligne);

Console.Write("la quantite tapee est : " + qte_comm);

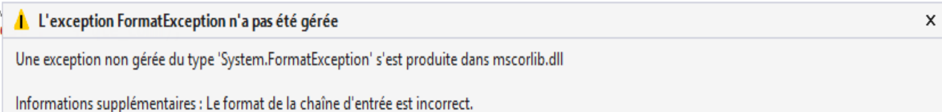
Console.ReadKey();
```




```
Console.WriteLine("tapez la quantite commandee du produit : ");
string ligne = Console.ReadLine();
int qte_comm = int.Parse(ligne);

Console.Write("la quantite tapee

Console.ReadKey();
```



La fonction ***TryParse( )*** des types **bool, int, short, byte, long, double, float, decimal et char**, permet d'éviter ce problème.
Syntaxe de base :
`bool resultat = type.TryParse(chaine, out variable)`
La fonction TryParse retourne la valeur **true** si la conversion s'est déroulée avec succès, sinon elle retourne la valeur **false**.

Exemple

```
Console.WriteLine("tapez la quantite commandee du produit : ");
string ligne = Console.ReadLine();
int qte_comm;
bool resultat  = int.TryParse(ligne, out qte_comm);
if(resultat == true)
    Console.Write("la quantite tapee est : " + qte_comm);
else
    Console.Write("la quantite tapee est incorrecte");
Console.ReadKey();
```

```
file:///C:/Users/s_elyahyaoui/Documents/Visual Studio 2015/Projects/CoursCSPr...
tapez la quantite commandee du produit :
1 1
la quantite tapee est incorrecte
```

REMARQUE

La structure de choix **`if / else`** sera détaillée dans un paragraphe un peu plus loin dans ce chapitre.